

## QNX PORTING KIT

OS Changer QNX™ Porting Kit is a C/C++ source-level virtualization technology that allows you to easily re-use your software developed using QNX APIs on another OS, while providing real-time performance. It eliminates the manual porting effort, saves money and shortens the time to market. OS Changer can also be used to simulate the QNX Interface on a host machine. OS Changer Interface connects to your existing application that was developed on QNX while the OS Abstractor Target Specific Module (specific to your target OS) provides the connection to the OS you are moving to.

### OPTIMIZED CODE GENERATION: OPTION ONE

- Legacy porting tool to easily import your QNX applications into AppCOE
- Perform your porting & simulation on Windows/Linux host machine with the provided GNU tools for x86
- Generate optimized QNX Interface code for your target, specific to your application
- Generate project files for your target IDE/tools environment
- Enable target profiling of the QNX Interface and of the application functions to collect valuable performance data and generate comparative performance reports
- Selectively optimize each QNX Interface function for performance based on its usage in your application
- Automatically generate initialization and configuration code based on the settings you chose in the GUI-based wizard

### FULL SOURCE PACKAGE GENERATION: OPTION TWO

- Use with your preferred IDE/tools instead of the AppCOE development environment
- Provides a Porting Kit in a source code format which contains all the QNX Interface functions for a specific target OS
- Requires manual configuration and initialization instead of using the AppCOE GUI-based wizard

## QNX PORTING KIT

### STEP ONE • Choose an option

#### Option One Optimized Code Generation

Your QNX App:  
Import using the legacy  
porting tool in AppCOE

Porting steps:  
Replace headers, Combine main ( )

Run/debug application  
using emulator on host

Configure target OS, Profiler,  
Interface Optimizer & system settings

Generate code  
for target OS

Output:  
• Unmodified application source  
• QNX Interface source

Native compiler

Output:  
• OS Abstractor API objects/library  
• QNX API objects/library  
• Application objects/libraries

Continue to **STEP TWO**

#### Option Two Full Source Package Generation

Your  
QNX App

Porting steps:  
Replace headers,  
Combine main ( )  
and Initialize app

Native  
compiler

Generate source  
package from AppCOE  
and perform manual  
configuration

Output:  
• OS Abstractor API library  
• QNX API library  
• Application objects/libraries

Continue to **STEP TWO**

### STEP TWO

Linker

Your  
QNX App  
executable

Download/run  
on your target OS

Generated Profiler  
data (optional)

View data using  
AppCOE Profiler

## Technical Highlights

### Includes a Process Feature

- > Port your application to a single or multiple processes utilizing the user shared region provided for your global variables
- > Create a new process by compiling the application separately or by launching it from your main application
- > Provides software-based process features, even if the underlying target OS does not offer support
- > Applications can pre-allocate heap memory during process creation
  - \* Set maximum limits regarding the amount of heap memory each application can use to prevent applications from using up all of the system memory and impacting other applications

### API Flexibility

- > OS Abstractor APIs also available for use in your QNX application
- > OS Changer QNX Interface can be used within a single or across multiple applications

### QNX Pooling

- > Applications can pool threads to increase platform robustness and performance by eliminating the overhead associated with actual task creation and task deletion at run-time

### Mission Critical Features

- > Applications have the ability to asynchronously recover from fatal software errors through a soft reset by rolling the stack back to the start of the application

### Highly Scalable

- > The AppCOE GUI-based wizard reads your application to custom generate optimized QNX Interface code that is specific to your application resulting in increased performance and reduction of memory footprint

### Target Hardware Independence

- > Products support any target hardware supported by your target OS architecture, including 32/64 bit & SMP/UP architectures

### In-house OS Support

- > Can easily be extended to support your in-house OS

## QNX Interface API Coverage & Target OS Support

You can find the supported QNX APIs here:

[https://www.mapusoft.com/wp-content/uploads/documents/Release\\_Notes-QNX-APIs.pdf](https://www.mapusoft.com/wp-content/uploads/documents/Release_Notes-QNX-APIs.pdf)

Below are the target operating systems supported by the OS Changer QNX Porting Kit:

Android®	LynxOS-178®	Nucleus®	µC/OS III™
eCOS®	micro-ITRON®	QNX Neutrino RTOS®	Unix®
Linux/POSIX	Freescale MQX®	RT Linux®	VxWorks®
LynxOS®	NetBSD®	Solaris®	In-House
LynxOS-SE®	ThreadX®		

- A free evaluation can be downloaded here:  
<http://mapusoft.com/downloads/>
- You can contact MapuSoft to request a license key for evaluation here:  
<http://mapusoft.com/contact>
- User manuals & technical documentation can be found here:  
<http://www.mapusoft.com/techdata/>
- For any technical or sales questions please submit a ticket at the MapuSoft support site here:  
<http://mapusoft.com/support/>